Augmented Memory: Capitalizing on Experience Replay to Accelerate *De Novo* Molecular Design

Jeff Guo[1,2], Philippe Schwaller[1,2]
[1]Laboratory of Artificial Chemical Intelligence (LIAC), Institut des Sciences et Ingénierie Chimiques,
Ecole Polytechnique Fédérale de Lausanne (EPFL), Lausanne, Switzerland
[2]National Centre of Competence in Research (NCCR) Catalysis,
Ecole Polytechnique Fédérale de Lausanne (EPFL), Lausanne, Switzerland
{jeff.guo,philippe.schwaller}@epfl.ch

Abstract

Sample efficiency is a fundamental challenge in *de novo* molecular design. Ideally, molecular generative models should learn to satisfy desired objectives under minimal oracle evaluations (computational prediction or wet-lab experiment). This problem becomes more apparent when using oracles that can provide increased predictive accuracy but impose a significant cost. Molecular generative models have shown remarkable sample efficiency when coupled with reinforcement learning, as demonstrated in the Practical Molecular Optimization (PMO) benchmark. Here, we propose a novel algorithm called Augmented Memory that combines data augmentation with experience replay. We show that scores obtained from oracle calls can be reused to update the model multiple times. We compare Augmented Memory to previously proposed algorithms and show significantly enhanced sample efficiency in an exploitation task and a drug discovery case study requiring both exploration and exploitation. Our method achieves a new state-of-the-art in the PMO benchmark which enforces a computational budget, and outperforms the previous best performing method on 19/23 tasks. The code is available at `https://github.com/schwallergroup/augmented_memory`.

1 Introduction

A quintessential task in any molecular discovery campaign is identifying promising candidate molecules amidst an enormous chemical space[1]. With the democratization of computing resources, computational oracles can be deployed to query larger chemical spaces in search of the desired property profile. The use of such oracles has enabled researchers to identify functional materials[2], therapeutics[3–5], and catalysts[6], thus accelerating chemical discovery. However, there is generally a trade-off between oracle predictive accuracy and inference cost, such that the computational budget imposes a pragmatic constraint. Provided a sufficiently sample efficient model, it is conceivable for wet-lab experiments to be the oracle itself, as enabled by a high-throughput experimentation platform. Correspondingly, designing computational workflows and algorithms that are performant under minimal oracle calls is widely beneficial to the field of molecular design.

Recent advancements in *de novo* molecular design have positioned generative methods as a complementary approach to traditional virtual screening[3,7]. Core advantages of these models include the ability to sample chemical space outside the training data and by coupling an optimization algorithm, goal-directed learning can be achieved[8]. Although the field is relatively nascent, molecular generative models have identified experimentally validated therapeutic molecules[4,5,9,10] and organocatalysts[6]. An important shared commonality between these success stories is the inclusion of relatively computationally expensive oracles that are optimized. In drug design, molecular docking is

Preprint. Under review.

frequently used while in catalyst and materials design, quantum mechanical properties are of interest. Correspondingly, many generative models proposed in recent years have competed to demonstrate accelerated optimization of these properties. However, the heterogeneity of the assessment protocols makes comparisons difficult. Recently, Gao et al.[11] propose the Practical Molecular Optimization (PMO) benchmark which assesses 25 molecular generative models across 23 tasks, enforcing a computational budget of 10,000 oracle calls. Their results show that REINVENT[12,13], a recurrent neural network (RNN)-based generative model operating on simplified molecular-input line-entry system (SMILES)[14] is, on average, the most sample efficient generative model. REINVENT[12,13] uses a policy-based reinforcement learning (RL) algorithm to optimize a reward function in a goal-directed approach. Recently, alternative algorithms have been proposed in the form of Best Agent Reminder (BAR)[15] and Augmented Hill Climbing (AHC)[16] which both introduce bias towards high rewarding molecules to improve sample efficiency. Other studies show that experience replay, where the highest rewarding molecules sampled are stored and replayed to the model, improves sample efficiency.[10,13] More recently, Bjerrum et al.[17] proposed Double Loop RL to take advantage of the non-injective nature of SMILES and the ease with which they can be augmented. By obtaining different SMILES sequences for the same molecule, oracle scores can be re-used to perform multiple updates to the Agent. Their results show accelerated learning while maintaining the diversity of results, an aspect missing in many proposed benchmarks.

Sample efficiency is a limiting factor to enabling more exploration of chemical spaces of interest, such as in drug discovery where the reward is sparse, i.e., finding a needle in the haystack. In this paper, we highlight the importance of experience replay in policy-based RL algorithms for molecular generation. We propose a novel algorithm called Augmented Memory that combines experience replay with SMILES augmentation. We further propose Selective Memory Purge which removes entries in the replay buffer with undesired chemical scaffolds and jointly address sample efficiency and diversity. The main contributions of this paper are:

- We explicitly highlight the importance of experience replay on the sample efficiency of REINVENT and all proposed algorithmic modifications.
- We propose a novel algorithm called Augmented Memory which significantly outperforms all previous algorithms in sample efficiency. This is demonstrated in an exploitation task and a drug discovery case study.
- We propose a method called Selective Memory Purge, which can be used in conjunction with Augmented Memory to generate diverse molecules while retaining enhanced sample efficiency.
- We expand the PMO benchmark[11] by adding Augmented Memory and BAR[15] implementations. We further add experience replay to the implemented version of AHC[16,18] for comparison. Our algorithm achieves a new state-of-the-art and outperforms the previous state-of-the-art, REINVENT[12,13], on 19/23 tasks.

2 Related Work

Goal-directed Molecular Design with Policy-based Reinforcement Learning. Molecular generation can be framed as a policy-based RL problem, where a base model (Prior) is trained on a general dataset and fine-tuned (Agent) to generate molecules with desired property profiles. Existing works that follow this paradigm include SMILES-based RNNs[12,13,19–22], generative adversarial networks (GANs)[23–27], variational autoencoders (VAEs)[4,28,29], graph-based models[15,30–32], and GFlowNets[33]. Other RL methods include using a policy network to choose favourable actions in a genetic algorithm[34] and learning a value function instead of a policy[35]. While all methods can generate valid molecules and the policy can be fine-tuned via RL, none of the previous methods jointly address sample efficiency and a reliable mechanism to mitigate mode collapse. We note that GFlowNets[33] by construction can achieve diverse sampling but are not as sample efficient as demonstrated in the PMO benchmark[11]. By contrast, SMILES-based models, particularly REINVENT[12,13], have been shown to be amongst the most sample efficient molecular generative models, even when compared to the newest proposed models[11]. Moreover, their ability to learn complex molecular distributions[36] and satisfy multi-parameter optimization (MPO) objectives has been shown in diverse benchmarks, such as GuacaMol[37], MOSES[38], and PMO[11]. Our proposed Augmented Memory algorithm builds on this observation and exploits the non-injective nature of SMILES.

Sample Efficiency in Molecular Design. Many existing policy-based RL works for molecular design are based on the REINFORCE[39] algorithm, particularly for models operating on SMILES. Algorithmic alternatives present a unifying theme of using biased gradients to direct the policy towards chemical space with high reward. Neil et al.[40] explored Hill Climbing (HC) and Proximal Policy Optimization (PPO)[40]. Similarly, Atance et al. introduced Best Agent Reminder (BAR)[15] which keeps track of the best agent and reminds the current policy of favorable actions. Thomas et al. introduced Augmented Hill Climbing (AHC)[16], a hybrid of HC and REINVENT's algorithm, which updates the policy at every epoch using only the top-k generated molecules and shows improved sample efficiency. However, sample efficiency by itself is not sufficient for practical applications of molecular generative models as one should aim to generate diverse molecules that satisfy the objective function. To address this limitation, Bjerrum et al. built directly on REINVENT and introduced Double Loop RL[17]. By performing SMILES augmentation, the policy can be updated numerous times per oracle call. Their results showed improved sample efficiency compared to AHC, while maintaining diverse sampling.

Experience Replay for Molecular Design. Experience replay was first proposed by Lin et al.[41] as a mechanism to replay past experiences to the model so that it can learn from the same experience numerous times. Two paradigms in RL are on-policy and off-policy where the model's actions are dictated by its current policy or a separate policy known as the behavior policy, respectively[42]. Experience replay is usually applied in off-policy methods as past experiences are less likely to be applicable to the current policy. In molecular design, experience replay has been proposed by Blaschke et al.[13,43] and Korshunova et al.[10] to keep track of the best molecules sampled so far, based on their corresponding reward. Notably, both applications of experience replay are for on-policy learning using the REINFORCE[39] algorithm and only Korshunova et al. empirically show its benefit in sparse reward environments. We note that a similar mechanism was proposed by Putin et al.[25] using an external memory.

3 Proposed Method: Augmented Memory

In this work, we extend the observations by Korshunova et al.[10] and explicitly show the benefit of experience replay in dense reward environments, i.e., most molecules give at least *some* reward, for on-policy learning given a static objective function. This static nature means that regardless of the current policy, high-rewarding molecules will always receive the same reward, which supports the efficacy of experience replay in the on-policy setting for molecular generation. Next, we combine elements of HC and SMILES augmentation with experience replay, and propose to update the policy at every fine-tuning epoch using the entire replay buffer. A reward shaping mechanism[44] is introduced by using these extremely biased gradients towards high rewarding chemical space which we show significantly improves sample efficiency. This section describes each component of Augmented Memory (Figure 1), which is capable of performing MPO.

Squared Difference Loss. The molecular generative model builds directly on REINVENT[12,13] and is an autoregressive SMILES-based RNN using long short-term memory (LSTM)[45] cells. The generative process is cast as an on-policy RL problem by defining the state space, S_t, and the action space, $A_t(s_t)$. Since REINVENT is a language model and samples tokens, S_t denotes every intermediate sequence of tokens leading up to the fully constructed SMILES and $A_t(s_t)$ are the token sampling probabilities at every intermediate state. $A_t(s_t)$ is controlled by the policy, π_θ, which is parameterized by the RNN. An assumption is that the SMILES (x) generation process is Markovian (Equation 1):

$$P(x) = \prod_{t=1}^{T} P(s_t \mid s_{t-1}, s_{t-2}, \ldots, s_1) \tag{1}$$

The Augmented Likelihood is defined as a linear combination between the Prior Likelihood and the scoring function, S, which returns a reward denoting the desirability of a given molecule and modulated by a hyperparameter sigma, σ (Equation 2). The Prior Likelihood term acts to ensure the generated SMILES are syntactically valid, and has been shown to empirically enforce reasonable chemistry[12,16].

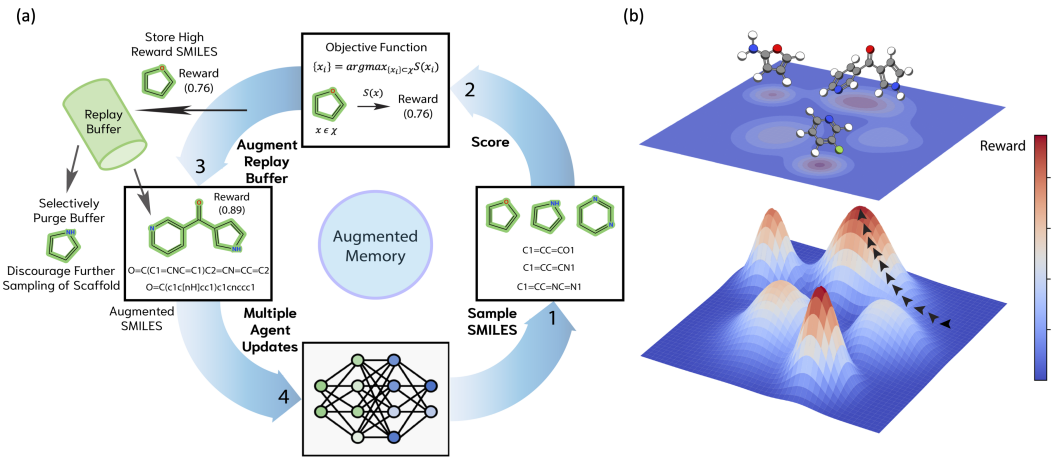

Figure 1: Augmented Memory. (a) The proposed method proceeds via four steps: 1. generate a batch of SMILES according to the current policy. 2. Compute the reward for the SMILES given the objective function. 3. Update the replay buffer to keep only the top K molecules. Optionally, remove molecules from the replay buffer to discourage further sampling of specific scaffolds. Perform SMILES augmentation of both the sampled batch and the entire replay buffer. 4. Update the Agent and repeat step 3 N times. (b) Schematic of the intended behavior. Augmenting the entire replay buffer and updating the Agent repeatedly directs chemical space exploration to areas of high reward.

$$\log \pi_{\theta_{\text{Augmented}}} = \log \pi_{\theta_{\text{Prior}}} + \sigma S(x) \tag{2}$$

The policy is directly optimized by minimizing the squared difference between the Augmented Likelihood and the Agent Likelihood given a sampled batch, B, of SMILES constructed following the actions, $a \in A^*$ (Equation 3):

$$L(\theta) = \frac{1}{|B|}\left[\sum_{a \in A^*} (\log \pi_{\theta_{\text{Augmented}}} - \log \pi_{\theta_{\text{Agent}}})\right]^2 \tag{3}$$

Taking the gradient of the loss function yields Equation 4:

$$\nabla_\theta L(\theta) = -2\frac{1}{|B|}\left[\sum_{a \in A^*} \log \pi_{\theta_{\text{Augmented}}} - \log \pi_{\theta_{\text{Agent}}}\right] \sum_{a \in A^*} \nabla_\theta \log \pi_{\theta_{\text{Agent}}} \tag{4}$$

Equivalency of the Squared Difference Loss to Policy Gradient Optimization. Minimizing the loss function described in Equation 3 is equivalent to maximizing the expected reward. To show this equivalency, we follow Fialková et al.[46] and start with the following objective, where R is the reward function (Equation 5):

$$J(\theta) = \mathbb{E}_{a_t \sim \pi_\theta}\left[\sum_{t=0}^{T} R(a_t, s_t)\right] \tag{5}$$

Following the REINFORCE[39] algorithm and applying the log-derivative trick yields Equation 6 for the gradient:

$$\nabla_\theta J(\theta) = \mathbb{E}_{a_t \sim \pi_\theta}\left[\sum_{t=0}^{T} R(a_t, s_t) \nabla_\theta \log \pi_\theta(a_t | s_t)\right] \tag{6}$$

Computing the expectation is intractable and is instead approximated using the mean of a sampled batch, B, of SMILES constructed by choosing actions, $a \in A^*$. Further noting that $\log \pi_\theta(a_t|s_t) = \log \pi_{\theta_{\text{Agent}}}$ yields Equation 7:

$$\nabla_\theta J(\theta) = \frac{1}{|B|}\left[\sum_{t=0}^{T}\sum_{a\in A^*} R(a_t, s_t)\nabla_\theta \log \pi_{\theta_{\text{Agent}}}\right] \tag{7}$$

Finally, the reward is defined as $R(a_t, s_t) = \log \pi_{\theta_{\text{Augmented}}} - \log \pi_{\theta_{\text{Agent}}}$. The corresponding gradient expression (Equation 7) is now equivalent to the gradient of the loss function (Equation 4) up to a constant factor. Further details on the derivation and algorithm is in Appendix K.

SMILES Augmentation. SMILES are non-injective and yield different sequence representations given a different atom numbering in the molecular graph, i.e., augmented SMILES. SMILES-based molecular generative models have taken advantage of this to train performant models under low-data regimes, e.g., by artificially increasing the dataset size via data augmentation[47], and to increase chemical space generalizability[48] by training a Prior model on augmented SMILES. Similar to Bjerrum et al.[17], we reuse scores obtained from the oracle to update the Agent multiple times by passing different augmented SMILES representations.

Experience Replay. Experience replay is implemented as a buffer that stores a pre-defined maximum number of the highest rewarding SMILES sampled so far (100 in this work). Usually, during each sampling, a subset of the buffer is replayed to the Agent[13]. In our proposed method, all SMILES in the buffer are augmented and using their corresponding reward, the Agent is updated multiple times according to the loss function given in Equation 3.

Selective Memory Purge. Blaschke et al.[43] introduced memory-assisted RL to enforce diverse sampling in REINVENT via diversity filters (DFs). During the generative process, the scaffolds of sampled molecules are stored in 'buckets' with pre-defined and limited size. Once a bucket has been fully populated, further sampling of the same scaffold results in zero reward. We incorporate this heuristic in our proposed method called Selective Memory Purge to enforce diversity. At every epoch, the replay buffer is purged of any scaffolds that are penalized by the DF. The effect is that each augmentation round only updates the Agent with scaffolds that still receive reward, preventing the Agent from becoming myopic and leading to sub-optimal convergence.

4 Results & Discussion

We designed three experiments to assess our method. First, we explicitly demonstrate the importance of experience replay and identify optimal parameters for Augmented Memory using the Aripiprazole Similarity experiment. Next, we benchmark its performance on the Practical Molecular Optimization (PMO)[11] benchmark containing 23 tasks. Lastly, we demonstrate the practical applicability of our method on a Dopamine Type 2 Receptor (DRD2) drug discovery case study.

Baselines. In experiments 1 and 3, the baseline algorithms include REINVENT[12,13], Augmented Hill Climbing (AHC)[16], Best Agent Reminder (BAR)[15], and Double Loop RL[17] as all algorithms are formulated using REINVENT's[12,13] loss function with shared hyperparameters. Thus, the experiments isolate the effect of SMILES augmentation and experience replay on sample efficiency. Moreover, SMILES-based models are strong baselines as demonstrated in the GuacaMol[37], MOSES[38], and PMO[11] benchmarks. In the PMO[11] benchmarking experiment, we compare our method's performance to diverse molecular optimization models. The Appendix includes further details on the dataset, hyperparameters, and ablation studies.

4.1 Aripiprazole Similarity

The aripiprazole similarity task is from the GuacaMol benchmark[37] and the objective is to successfully sample aripiprazole. This experiment was used to demonstrate the importance of experience replay and compare Augmented Memory to existing policy-based algorithms. As the code for Double Loop RL is not released, we took the values reported in their paper which holds as the method was also built directly on REINVENT[12,13], uses the same pre-trained Prior, and hyperparameters. Moreover,

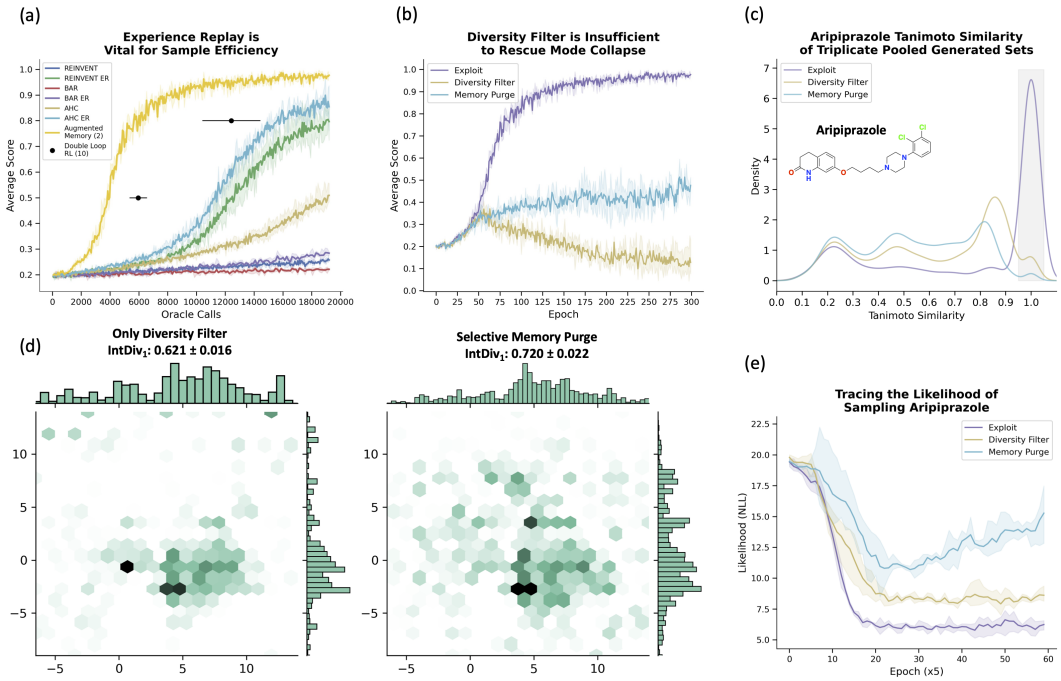

Figure 2: Augmented Memory and Selective Memory Purge significantly improve sample efficiency and enable diverse sampling. The shaded region represents the minimum and maximum scores across triplicate runs. (a) Comparing sample efficiency of on-policy algorithms. Experience Replay (ER) improves all base algorithms. The values for Double Loop RL [17] are taken from the original paper as the code is not released. The black dots are the mean at 0.5 and 0.8 and the standard deviation across triplicate runs. (b) The average score for aripiprazole similarity. In the Diversity Filter and Memory Purge experiments, scores of 0 are given if the Agent repeatedly samples the same scaffold. (c) Pooled Tanimoto similarities. Memory purge rediscovers aripiprazole and has a flatter distribution, suggesting increased exploration. (d) UMAP [49] of Morgan fingerprints [50] and IntDiv1 [38] metric showing qualitatively and quantitatively increased exploration using Memory Purge. The plots were generated using ChemCharts [51] (e) The negative log-likelihood of sampling aripiprazole across the full generative experiments.

in the studies presenting AHC [16] and BAR [15], experience replay was not used but we provide an implementation and further compare their performance.

Experience Replay is Vital for Sample Efficiency. We first identified the optimal number of augmentation rounds for Augmented Memory as two for training stability. Increasing the augmentation rounds can further improve sample efficiency but can lead to mode collapse (Appendix A). Next, we compare REINVENT [12,13], AHC [16], BAR [15], and Double Loop RL [17] with our method. Figure 2 plots the number of oracle calls to explicitly highlight the computational budget. Augmented Memory significantly outperforms all other algorithms and reaches a score of 0.8 with 6,144 oracle calls (average over 100 replicates). Double Loop RL [17] uses experience replay and is the second most sample efficient algorithm and reaches a score of 0.8 after 12,416 $\pm$ 1,984 oracle calls (as stated in their paper), which is twice the number of oracle calls required compared to our method. Moreover, the key observation we convey is that experience replay improves upon the base algorithm in all cases (Figure 2). For example, AHC [16] with the newly implemented experience replay reaches a score of 0.8, but with more than 2.5x the oracle calls (15,552). Our observations around experience replay are supported by previous works [10,13]. Finally, we show that augmentation is crucial for the enhanced sample efficiency in Appendix D.

Selective Memory Purge Enables Diverse Sampling while Retaining Efficiency. Figure 2 demonstrates the enhanced sample efficiency of Augmented Memory but real-world applications of molecular generative models require the ability to sample diverse solutions. While aripiprazole is inherently an exploitation task, it can be framed as an exploration task if the goal is rephrased as: rediscover the

target molecule and generate similar molecules. Using this formulation, we design experiments to prove that Augmented Memory can achieve diverse sampling. Figure 2 shows the training plot across three methods: pure exploitation where diversity is not enforced, exploration using a diversity filter (DF)[43], and Selective Memory Purge. In the pure exploitation scenario, aripiprazole is rediscovered quickly (score of 1.0). In the DF experiment where a score of 0 is assigned for scaffolds sampled more than 25 times, mode collapse is observed (Figure 2). By contrast, Selective Memory Purge maintains a moderate average score. The results from triplicate experiments were pooled to investigate the density of aripiprazole similarities (Figure 2). As expected, in the pure exploitation scenario, most molecules are aripiprazole (Tanimoto similarity of 1.0). DF and Selective Memory Purge both enforce a wider distribution of similarities, but to varying degrees. In the shaded region (rediscovery score), Selective Memory Purge only shows a small density relative to DF. Moreover, Selective Memory Purge shows a flatter distribution of similarities. These observations demonstrate that Selective Memory Purge rediscovers the target molecule and enforces increased exploration compared to DF. To investigate this further, the Morgan fingerprints[50] of the same pooled dataset were embedded using Uniform Manifold Approximation and Projection (UMAP)[49] to visualize the chemical space. Qualitatively and quantitatively, Selective Memory Purge covers a larger chemical space (Figure 2). The internal diversity (IntDiv1) metric was calculated as proposed in the MOSES benchmark[38], and measures the diversity within a set of generated molecules. Finally, we save the Agent states at every 5 epochs across the entire generative run and trace the NLL of sampling aripiprazole (Figure 2). It is evident that Selective Memory Purge can discourage sampling of the target molecule more effectively than only using a DF. Importantly, the NLL also diverges, suggesting that the Agent is increasingly moving to chemical space dissimilar to aripiprazole as the generative experiment progresses.

4.2 Practical Molecular Optimization (PMO) Benchmark

Table 1: Performance of Augmented Memory, REINVENT[12,13], AHC[16], and BAR[15] on the PMO benchmark[11]. The mean and standard deviation of the AUC Top-10 is reported. The values obtained for REINVENT differ slightly from the PMO paper as we performed 10 independent runs compared to 5. Best performance is bolded and is relative to all models in the benchmark. * denotes superior performance to REINVENT but not overall, compared to other models in the benchmark. We note however, that we take the AUC Top-10 values for the other models as is from the PMO paper. If they were re-run with 10 different seeds (instead of 5), the values may decrease as was observed for REINVENT.

Benchmark Task	**Augmented Memory**	**REINVENT**	**AHC Replay**	**BAR Replay**	**AHC**	**BAR**
albuterol_similarity	**0.913 ± 0.009**	0.871 ± 0.031	0.792 ± 0.030	0.700 ± 0.024	0.745 ± 0.024	0.633 ± 0.031
amlodipine_mpo	**0.691 ± 0.047**	0.657 ± 0.025	0.596 ± 0.023	0.538 ± 0.019	0.578 ± 0.012	0.523 ± 0.006
celecoxib_rediscovery	**0.796 ± 0.008**	0.717 ± 0.048	0.697 ± 0.029	0.563 ± 0.043	0.583 ± 0.070	0.437 ± 0.025
deco_hop	0.658 ± 0.024	0.672 ± 0.052	0.650 ± 0.030	0.589 ± 0.010	0.632 ± 0.032	0.579 ± 0.008
drd2	**0.963 ± 0.006***	0.939 ± 0.012	0.913 ± 0.011	0.916 ± 0.012	0.912 ± 0.009	0.899 ± 0.027
fexofenadine_mpo	**0.859 ± 0.009**	0.783 ± 0.021	0.747 ± 0.004	0.708 ± 0.010	0.749 ± 0.005	0.692 ± 0.009
gsk3b	**0.881 ± 0.021**	0.870 ± 0.026	0.819 ± 0.025	0.744 ± 0.021	0.800 ± 0.021	0.686 ± 0.068
isomers_c7h8n2o2	0.853 ± 0.087	0.856 ± 0.042	0.682 ± 0.037	0.741 ± 0.064	0.631 ± 0.084	0.713 ± 0.058
isomers_c9h10n2o2pf2cl	**0.736 ± 0.051***	0.641 ± 0.038	0.276 ± 0.133	0.612 ± 0.054	0.191 ± 0.096	0.508 ± 0.066
jnk3	**0.739 ± 0.110**	0.723 ± 0.147	0.649 ± 0.056	0.555 ± 0.089	0.616 ± 0.092	0.511 ± 0.092
median1	0.326 ± 0.013	0.368 ± 0.011	0.346 ± 0.008	0.286 ± 0.007	0.338 ± 0.014	0.269 ± 0.011
median2	**0.291 ± 0.008***	0.279 ± 0.005	0.273 ± 0.005	0.218 ± 0.008	0.265 ± 0.005	0.199 ± 0.006
mestranol_similarity	**0.750 ± 0.049**	0.637 ± 0.041	0.599 ± 0.031	0.463 ± 0.027	0.561 ± 0.022	0.444 ± 0.017
osimertinib_mpo	**0.855 ± 0.004**	0.836 ± 0.007	0.810 ± 0.003	0.789 ± 0.012	0.809 ± 0.002	0.792 ± 0.004
perindopril_mpo	**0.613 ± 0.015**	0.561 ± 0.019	0.487 ± 0.012	0.468 ± 0.012	0.482 ± 0.008	0.455 ± 0.011
qed	**0.942 ± 0.000**	0.941 ± 0.000	0.941 ± 0.000	0.939 ± 0.003	0.941 ± 0.000	0.932 ± 0.007
ranolazine_mpo_mpo	**0.801 ± 0.006**	0.768 ± 0.008	0.721 ± 0.00	0.704 ± 0.017	0.722 ± 0.008	0.700 ± 0.021
scaffold_hop	**0.567 ± 0.008**	0.556 ± 0.019	0.535 ± 0.007	0.477 ± 0.010	0.525 ± 0.008	0.464 ± 0.005
sitagliptin_mpo	**0.284 ± 0.050***	0.049 ± 0.067	0.022 ± 0.008	0.126 ± 0.049	0.028 ± 0.011	0.070 ± 0.020
thiothixene_rediscovery	**0.550 ± 0.041***	0.531 ± 0.036	0.519 ± 0.012	0.396 ± 0.011	0.467 ± 0.032	0.347 ± 0.013
troglitazone_rediscovery	**0.540 ± 0.048**	0.428 ± 0.028	0.409 ± 0.020	0.301 ± 0.007	0.371 ± 0.019	0.279 ± 0.007
valsartan_smarts	0.000 ± 0.000	0.091 ± 0.273	0.000 ± 0.000	0.000 ± 0.000	0.000 ± 0.000	0.000 ± 0.000
zaleplon_mpo	**0.394 ± 0.026**	0.269 ± 0.083	0.072 ± 0.032	0.319 ± 0.033	0.047 ± 0.013	0.294 ± 0.014
Sum of AUC Top-10 (↑)	**15.002**	14.016	12.555	12.152	11.993	11.426
PMO Rank ($n/29$)	**1**	2	7	9	11	14

The main motivation of our method is to improve sample efficiency. This would enable molecular generative models to explicitly optimize more expensive oracles which can afford increased predictive accuracy. We benchmark our method on the PMO benchmark proposed by Gao et al.[11] which restricts the number of oracle calls to 10,000 and encompasses 23 tasks. The metric used is the Area Under the Curve (AUC) for the top 10 molecules. We note that Thomas et al.[18] proposed a modified AUC Top-10 metric that incorporates diversity, but we omit comparison as the formulation can be subjective. The current Top AUC-10 metric assesses sample efficiency which is our focus. In the original PMO paper, REINVENT[12] (with experience replay) is the most sample efficient model. We compare our method directly to REINVENT, BAR[15], and AHC[16] which reports improved sample efficiency compared to REINVENT and is open-sourced. We also add experience replay to BAR and AHC to highlight its importance for sample efficiency. For a more statistically convincing comparison, we perform 10 independent runs (using 10 different seeds) compared to 5 used in the original PMO[11] paper as the authors benchmarked 25 models, which imposed a significant computational cost. The optimal hyperparameters for REINVENT and AHC were used as provided in the PMO repository. We perform hyperparameter optimization for BAR following the PMO protocol (Appendix G) and Augmented Memory was run using REINVENT's optimal hyperparameters. The results show Augmented Memory significantly outperforms all methods and achieves superior performance to REINVENT across 19/23 tasks (14/23 compared to all models in PMO[11]) (Table 1). Moreover, the results reinforce the importance of experience replay as it improves the sample efficiency of both BAR and AHC, although neither outperform REINVENT. Finally, in the PMO paper[11], models were ranked based on the sum of the total AUC Top-10 and adjacently ranked models typically differ by 0.3-0.5. Augmented Memory outperforms REINVENT by 0.986 AUC Top-10 and yields a new state-of-the-art performance on the PMO benchmark.

4.3 Dopamine Type 2 Receptor (DRD2) Case Study

To prove that Augmented Memory can perform MPO, we formulate a case study to generate potential dopamine type 2 receptor (DRD2) inhibitors[52] by explicitly optimizing molecular docking scores (Figure 3). For accessibility and reproducibility, we use the open-source AutoDock Vina[53] for docking. A well-known failure mode of docking algorithms is they reward lipophilic molecules, e.g., possessing many carbon atoms, which can be promiscuous binders[54,55]. Bjerrum et al.[17] consider this and enforced molecules to possess a molecular weight (MW) < 500 Da but this is insufficient in preventing exploitation of the docking algorithm as we show in Appendix E. Following Guo et al.[56], we design the MPO as follows: MW < 500 Da, maximize QED[57], and minimize the Vina docking score, for chemical plausibility. AutoDock Vina is a relatively expensive oracle and we impose a computational budget of 9,600 oracle calls, similar to the 10,000 oracle calls enforced in the PMO[11] benchmark. We compare Augmented Memory, REINVENT[12,13], AHC[16], and BAR[15] as the optimization algorithms. To mimic a real-world drug discovery pipeline that discards unpromising molecules, we pool the results from triplicate experiments with the following filter: MW < 500 Da, QED > 0.4 (the DRD2 drug molecule, risperidone, has a QED of 0.66), and Vina docking score < -9.4 (risperidone's score). Figure 3 shows the docking scores distribution with the number of molecules passing the filter and the IntDiv1[38] score annotated. Firstly, experience replay improves all base algorithms, further reinforcing its importance. Secondly, all algorithms with the exception of Augmented Memory perform similarly. Compared to AHC with experience replay, which is the second most sample efficient algorithm, Augmented Memory generates over 2,000 more molecules with a better docking score than risperidone, with a small trade-off in diversity (IntDiv1[38] of 0.801). We emphasize that AHC with experience replay only generates 1,667 molecules passing the filter. To further prove the optimization capability, Figure 3 shows a contour plot of the QED-Vina score distribution for Augmented Memory and AHC with experience replay. It is clear that the joint QED-Vina score distribution for Augmented Memory is shifted to higher QED values and lower Vina scores. The black dot is risperidone and the bulk density of AHC does not possess a better docking score. Finally, Figure 3 shows an example binding pose of a molecule generated using Augmented Memory. We highlight that the chemical plausibility of the structure is enforced precisely because MW and QED are also included in the MPO objective, thus representing a more realistic case study.

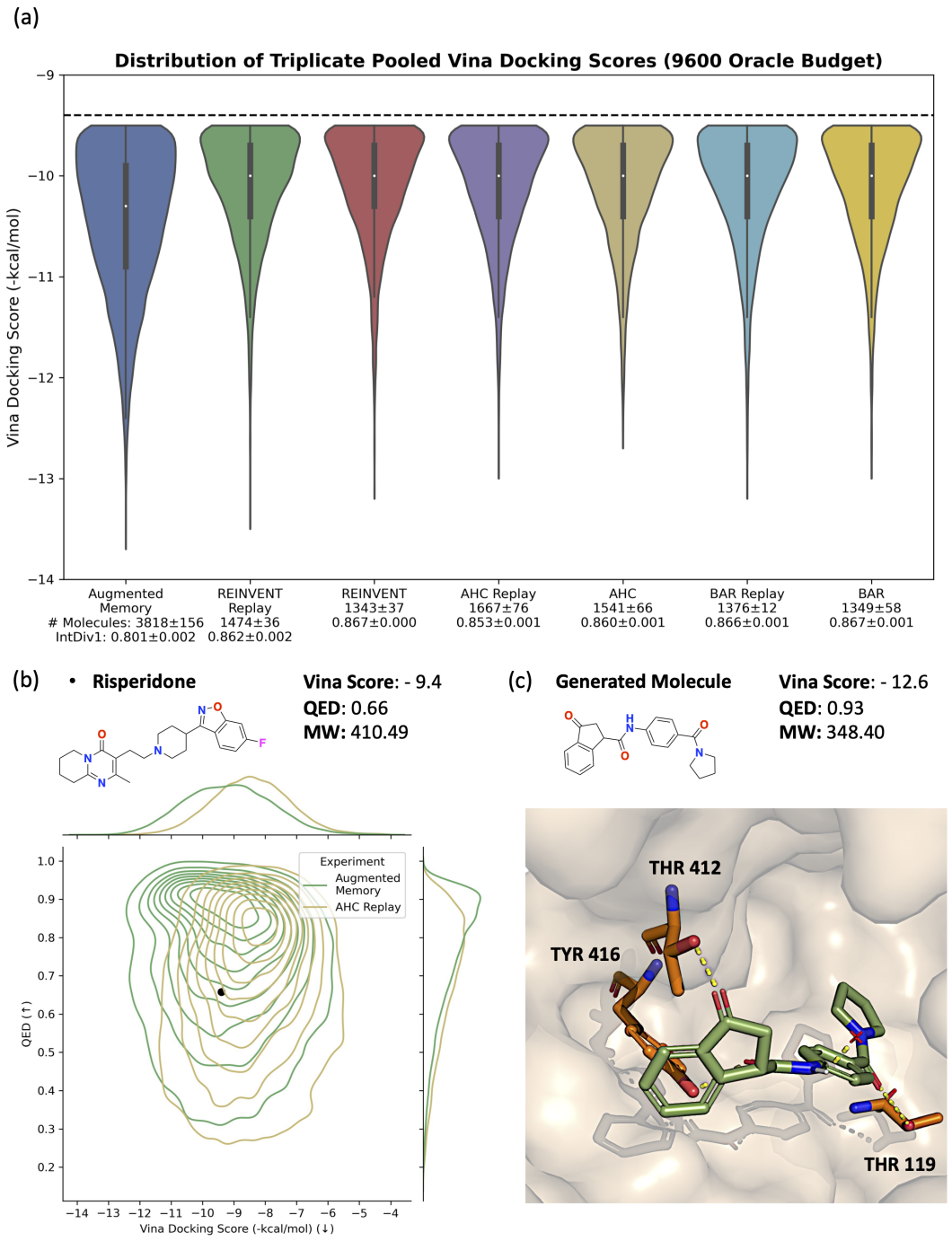

Figure 3: Dopamine type 2 receptor (DRD2) molecular docking case study. PDB ID: 6CM4. (a) Docking scores distribution of all compared algorithms. (b) Augmented Memory jointly optimizes QED and Vina docking score, demonstrating the ability to perform MPO. (c) Binding pose of a generated molecule using Augmented Memory. The three components in the objective function: MW < 500, QED, and Vina docking score are all optimized.

5 Conclusion

In this work, we explicitly show that experience replay is vital for sample efficiency. We propose Augmented Memory which capitalizes on this observation and applies SMILES augmentation on the replay buffer to update the Agent multiple times per oracle call. Compared to existing algorithms, Augmented Memory significantly improves sample efficiency and is able to generate diverse molecules using the newly proposed Selective Memory Purge heuristic. We benchmark Augmented Memory on the PMO benchmark [11] and achieve a new state-of-the-art performance, outperforming the previous state-of-the-art on 19/23 tasks and by a total sum of 0.986 AUC Top-10. Next, we show the practical application of Augmented Memory by mimicking a more realistic drug discovery task. Our method significantly outperforms existing algorithms, as assessed by the property profile of the generated molecules, and can perform MPO. Moreover, we note that in particularly sparse reward landscapes [10], the enhanced sample efficiency of Augmented Memory may be diminished as it becomes more difficult to populate the replay buffer with high rewarding molecules. Future work will investigate this scenario thoroughly and algorithmic modifications to couple additional local chemical space exploration [58] around high rewarding molecules may better handle sparsity. This work opens up future integration of Augmented Memory with curriculum learning [59,60], the use of more expensive oracles given a limited computational budget, and provides further insights into experience replay for molecular generative models.

A Tolerability to Augmentation Rounds

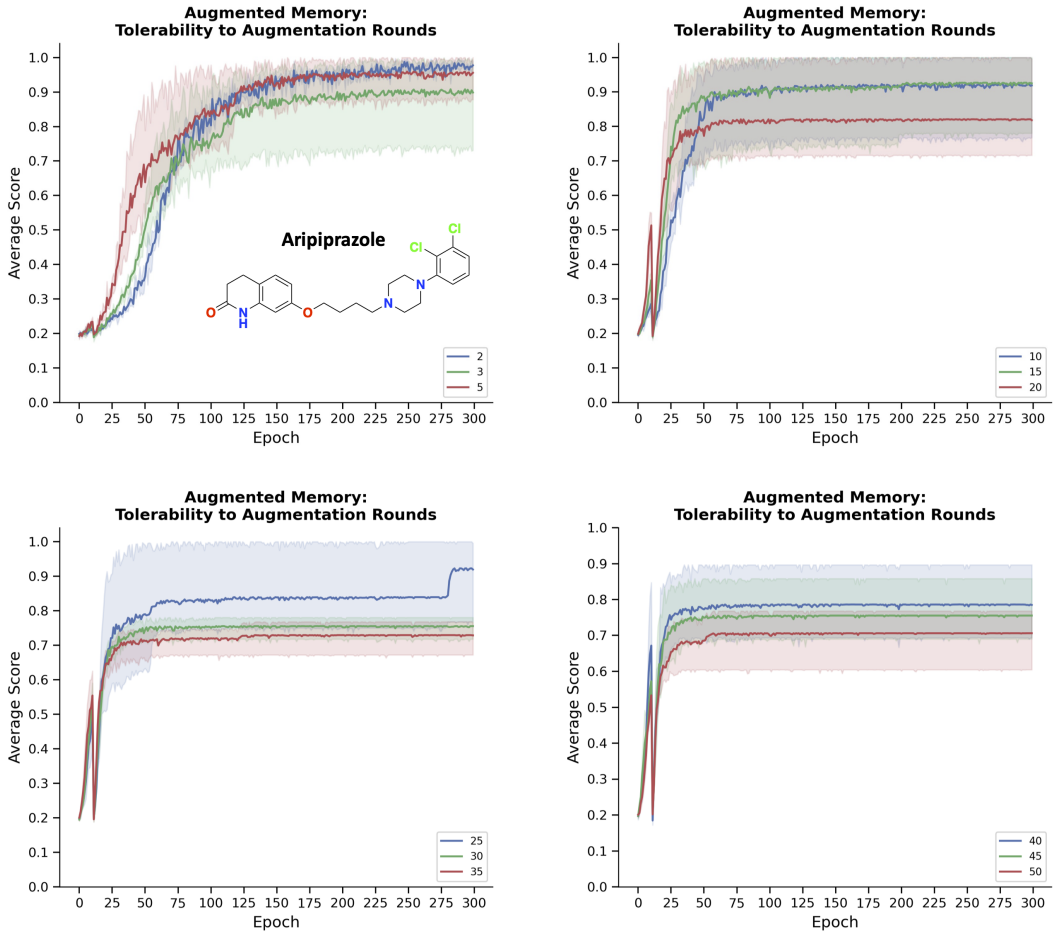

Figure 4: Identifying the optimal augmentation rounds using Aripiprazole Similarity. The shaded region represents the minimum and maximum scores across triplicate runs.

Similar to Esben et al.[17] in their proposed Double Loop RL algorithm, increasing the number of augmentation rounds increases susceptibility to mode collapse (Figure 4). We used the Aripiprazole Similarity task to perform a grid optimization and found two rounds to be optimal for stability. At three rounds, mode collapse is already observed with triplicate runs.

B Pure Exploitation: Robustness of 2 Augmentation Rounds

Table 2: Robustness experiments: stability of two augmentation rounds. 100 replicates of Aripiprazole Similarity was performed using 2 augmentation rounds and the epoch number to reach various average scores are presented. The values for Double Loop RL[17] are for 10 augmentations which the authors state to be most stable

Average Score	Mean Epochs	Double Loop RL Mean Epochs
0.5	65 ± 6	93 ± 9
0.8	99 ± 23	194 ± 31
0.9	122 ± 19	did not report

Initial screening experiments identified two augmentation rounds to be optimal for training stability. We envisioned in **pure exploitation** scenarios where **Selective Memory Purge** is not used, mode

collapse may be possible. The rationale being that the Agent is reinforced on the same replay buffer molecules. In the case where the entire replay buffer contains very similar or identical molecules, mode collapse may occur. **This is not an issue when using Selective Memory Purge as entries in the replay buffer would be removed, thus preventing the entire buffer containing the same molecules. We verified this statement by performing 100 replicates of Aripiprazole Similarity using Selective Memory Purge. All replicates rediscovered Aripiprazole, indicating no mode collapse at sub-optimal minima.**

In most practical applications of molecular generative models, Selective Memory Purge should be used to achieve both exploration and exploitation. However, for full transparency, we report the stability of our proposed method in a pure exploitation scenario. The following insights will be informative if prospective users only want to generate one optimal solution in their generative experiment or want to reproduce the Aripiprazole Similarity experiment. To preemptively prevent mode collapse, we introduce "mode collapse guard" that purges the replay buffer if 50 percent of the buffer contains the exact same reward. For statistical rigour, we perform 100 replicates of Aripiprazole Similarity and present the results in Table 2. We follow Bjerrum et al.[17] and present statistics on the epochs it takes to reach various average scores (average Tanimoto similarity of the batch of sampled molecules to aripiprazole) of 0.5, 0.8, and 0.9. The results support the stability of our method even in pure exploitation scenarios. The "mode collapse guard" was activated 16 times across 100 replicates and in all cases except 4, prevents mode collapse. The exceptions failed to rediscover aripiprazole (mode collapse at a Tanimoto similarity of 0.88). In practical applications, the experiment can be monitored and restarted from a check-point state. Moreover, we comment on the number of epochs it takes to reach an average score of 0.8 and 0.9 which are, in both cases, much lower than Double Loop RL[17]. In particular, even at three standard deviations from the mean, the epochs it takes our method is faster than Double Loop RL. We further note that it is unclear if running their algorithm for 100 replicates would still be stable as it is not open-sourced. Overall, in pure exploitation scenarios, mode collapse can be observed but is not a problem if Selective Memory Purge is used. We highlight the practical usage of Augmented Memory with Selective Memory Purge to generate diverse molecules on the DRD2 case study in the main results and in Appendix E. Our method is consistently stable.

We end this section by further discussing the choice to use a "mode collapse guard" and why it is not just to make our method look more robust. The insights in this section are for transparency on our method but also positions the work for future integration with curriculum learning (CL)[59,60]. In curriculum learning, a complex objective function is decomposed into simpler sequential tasks to accelerate the learning process. It may be advantageous to enforce pure exploitation for intermediate tasks as the objective is to learn as quickly as possible this task before moving to the successive task. In this case, enforcing diversity could be counterproductive. While mode collapse is rare, as discussed above, its possibility decreases the robustness of a direct CL integration. Leveraging the insights from this section, a lenient Selective Memory Purge (allowing a larger number of identical scaffolds to be sampled before penalization) can be applied on top of CL to retain Augmented Memory's sample efficiency while also capitalizing on benefits of CL.

C Buffer Size Experiments and Reinforcing with Only Experience Replay

As Augmented Memory revolves around exploiting experience replay, we investigate the efficacy of our method when using different buffer sizes (Figure 5). We again use the Aripiprazole Similarity task to assess the proposed changes. Interestingly, with the exception of a buffer size of 25, minimal difference is observed between buffer sizes. We posit that a buffer size of 25 is more susceptible to mode collapse as it is increasingly likely that the stored molecules are all identical or similar relative to having a larger buffer size. Conversely, our initial hypothesis was that a larger buffer size would decrease sample efficiency. The rationale is that relatively low rewarding molecules may be stored in the buffer and reinforcing on these low rewarding molecules could be counterproductive. Following experiments (Figure 5), this was not the case, at least for the Aripiprazole Similarity task. Given that the differences in the buffer sizes result in minimal difference and that our hypothesis may be true for other objective functions, we decided to use a buffer size of 100 for main result experiments. Next, we were curious if reinforcing the Agent with only the molecules in the replay buffer would be possible. In these experiments, the sampled molecules in a given epoch were only used to reinforce the Agent once and no augmented forms were used to further reinforce the Agent. Interestingly,

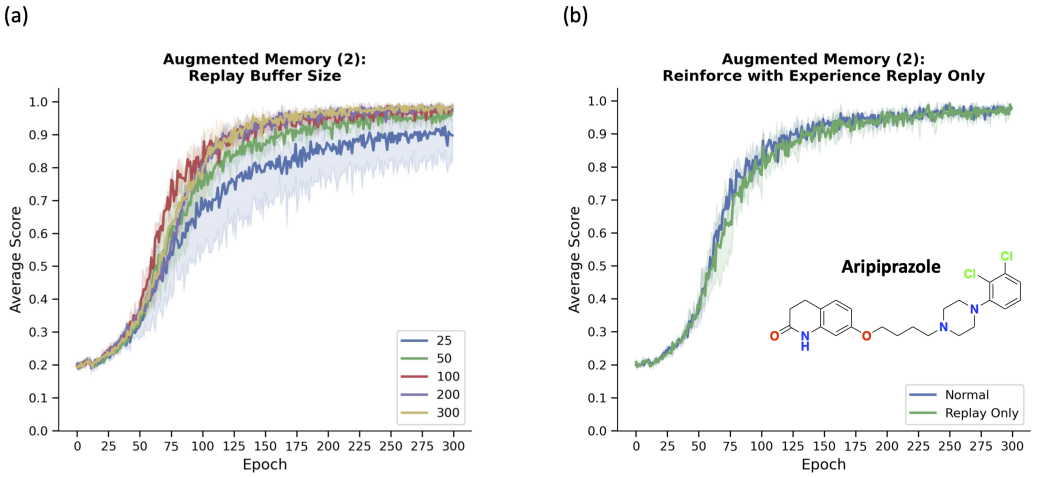

Figure 5: Investigating changes in the replay buffer size and reinforcing the Agent only with molecules stored in the replay buffer. The shaded region represents the minimum and maximum scores across triplicate runs.

minimal difference is observed again (Figure 5). Since the performance is similar, we hypothesize that using augmented forms of the sampled molecules would act to mitigate against mode collapse. This is in agreement with insights from Arús-Pous et al.[48] that posit SMILES augmentation acts as a regularizer. Therefore, all main result experiments were performed using augmented SMILES from the sampled batch and the buffer.

D Ablation Study: SMILES Augmentation is a Regularizer

Table 3: Stability without SMILES augmentation. 100 replicates of Aripiprazole Similarity was performed using 2 augmentation rounds (but without SMILES augmentation) and the epoch number to reach various average scores are presented. Failed runs did not reach the average score threshold. The epoch numbers for the runs with augmentation are shown in parenthesis for comparison.

Average Score	Mean Epochs	Failed Runs
0.5	$69(65) \pm 8(6)$	0
0.8	$119(99) \pm 41(23)$	3
0.9	$159(122) \pm 47(19)$	14

The results in the main text show that experience replay is vital for sample efficiency. In this section, the question we answer is: "can we just perform multiple rounds of Agent update with the entire replay buffer without SMILES augmentation?" If yes, then the benefits of Augmented Memory can be attributed to simply experience replay. The experimental design is as follows: using the Aripiprazole Similarity experiment, perform two rounds of Agent update using the entire buffer (size of 100) without SMILES augmentation. This mirrors the optimal parameters of Augmentation Memory of two augmentation rounds and a buffer size of 100. For statistical rigour, we perform 100 replicates and present the results in Table 3. Compared to Augmented Memory, the average epochs it takes to reach an average score of 0.5, 0.8, and 0.9 is higher (values with augmentation are shown in parentheses and is from Table 2). Importantly, the standard deviation is also much higher, suggesting instability in the runs. This is further supported by some runs not reaching the 0.8 and 0.9 average score thresholds. While monitoring the sampling, we notice that the Agent repeatedly samples the same SMILES, indicating mode collapse. From a probabilistic perspective, the Agent negative log-likelihoods (NLLs) become focused on the replay buffer sequences, suggesting token-level memorization. These insights are supported by previous work from Arús-Pous et al.[48] which explored the effect of SMILES augmentation on the Prior's NLL on the training data. Specifically, they found that training a Prior without SMILES augmentation can cause token-level memorization,

such that the NLL for the specific SMILES sequences in the training data are low. This decreases the generalizability of the trained Prior. Bjerrum et al.[17] in their Double Loop RL work also posit that reinforcing the Agent on augmented SMILES prevents sequence-wise mode collapse. Our results are in agreement and we show that SMILES augmentation is necessary to ensure the efficacy of Augmented Memory and is itself a regularizer.

E Dopamine Type 2 Receptor (DRD2) Case Study: Exploiting AutoDock Vina

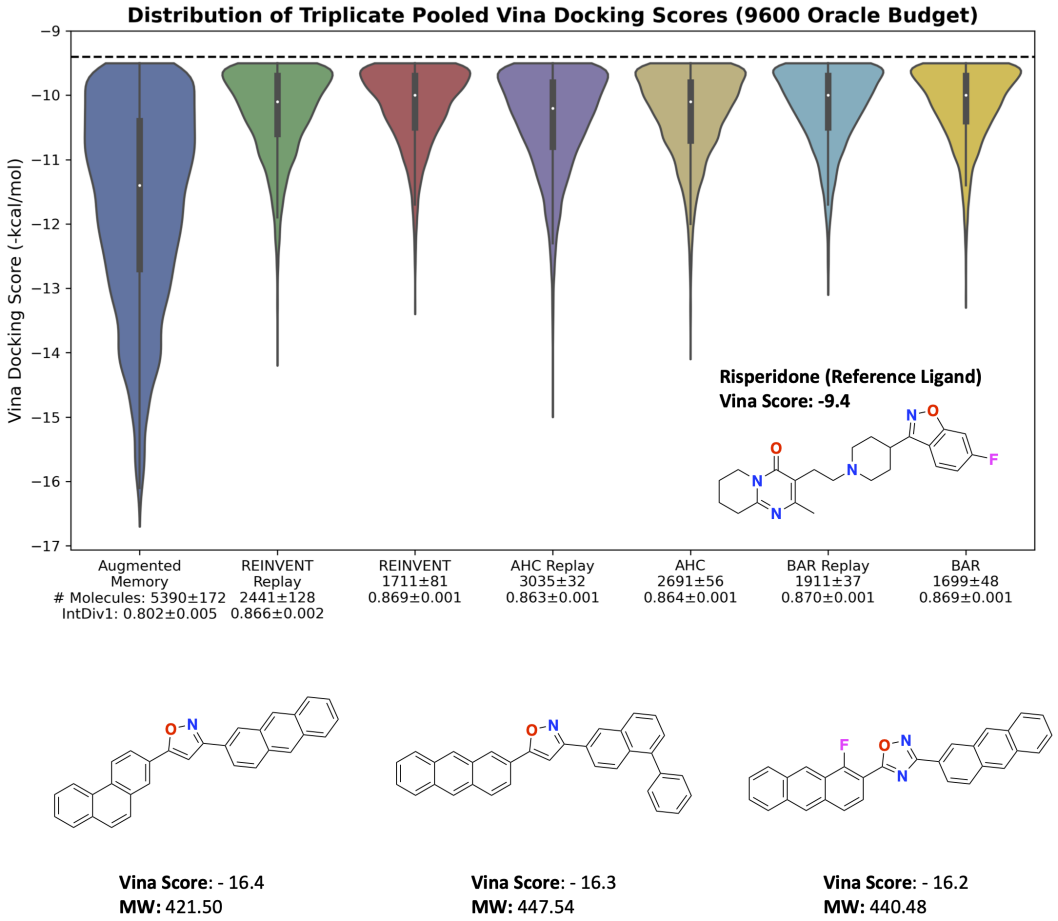

Figure 6: Dopamine type 2 receptor (DRD2) case study using the objective function: molecular weight < 500 Da and minimize Vina docking score. Augmented Memory significantly outperforms other algorithms. The generated molecules, however, are not realistic and shows that Augmented Memory can exploit objective functions in a sample efficient manner.

This section elaborates on the statement that the experimental design of Bjerrum et al.[17] in their Double Loop RL work is insufficient in preventing AutoDock Vina[53] exploitation. Specifically, the drug discovery case study to design potential dopamine type 2 receptor (DRD2) inhibitors was performed using the following objective function: molecular weight (MW) < 500 Da, maximize QED, and minimize docking score. This is in contrast to the objective function proposed by Bjerrum et al.[17]: molecular MW < 500 Da and minimize docking score. We perform a set of experiments comparing the sample efficiency of alternative algorithms including REINVENT[12,13], Best Agent Reminder (BAR)[15], and Augmented Hill Climbing (AHC)[16] using this simplified objective function. Similar to the main result experiments, we ran all experiments enforcing an oracle budget of 9,600 calls and show the distribution of Vina scores from triplicate pooled runs. The following filter was applied: MW < 500 Da and Vina score < -9.4 (the Vina score of the reference drug molecules, risperidone). It is evident that Augmented Memory significantly outperforms other algorithms, generating drastically

better docking scores and more molecules passing the filter. Moreover, experience replay improves the performance of all base algorithms. However, we investigate the property profile of the generated molecules in the Augmented Memory experiments and show that the Agent exploits AutoDock Vina in rewarding lipophilic molecules, i.e., all top scoring molecules have extensive aromatic carbon rings (Figure 6). These molecules, while possessing excellent Vina scores, are not realistic. As we emphasize the usability of Augmented Memory on more realistic case studies to encourage practical applications, we show the set of experiments which also enforce QED[57] in the main results. QED ensures generated molecules are "drug-like". We end this section by emphasizing that the ability of Augmented Memory to exploit AutoDock Vina is not a weakness and rather, further proves its ability for sample efficient optimization.

F Aripiprazole and DRD2 Prior and Hyperparameters

The *random.prior.new* pre-trained Prior was used from the REINVENT 2.0[13] repository which was trained on ChEMBL[61]. We note that for the Aripiprazole Similarity experiment, this enables direct comparison to Double Loop RL[17] as the authors also used the same Prior. The hyperparameters used for Experiment 1: Aripiprazole Similarity and Experiment 3: Dopamine Type 2 Receptor (DRD2) are presented in Table 4.

Table 4: Hyperparameters (default) used in Experiment 1: Aripiprazole Similarity and Experiment 3: Dopamine Type 2 Receptor (DRD2).

Algorithm	Sigma (σ)	Batch Size	Learning Rate	k	Alpha (α)
Augmented Memory	128	64	0.0001	N/A	N/A
REINVENT[12,13]	128	64	0.0001	N/A	N/A
Augmented Hill-Climbing (AHC)[16]	60	64	0.0001	0.5	N/A
Best Agent Reminder (BAR)[15]	1	64	0.0001	N/A	0.5

G Practical Molecular Optimization (PMO) Hyperparameters

Table 5: Hyperparameters used in Experiment 2: Practical Molecular Optimization (PMO)[11] Benchmark.

Algorithm	Sigma (σ)	Batch Size	Learning Rate	k	Alpha (α)
Augmented Memory	500	64	0.0005	N/A	N/A
REINVENT[12,13]	500	64	0.0005	N/A	N/A
Augmented Hill-Climbing (AHC)[16]	120	256	0.0005	0.25	N/A
Best Agent Reminder (BAR)[15]	1000	64	0.0005	N/A	0.25

The hyperparameters used for the Practical Molecular Optimization (PMO)[11] benchmark is presented in Table 5. The hyperparameters provided in the PMO repository for REINVENT[12,13] and AHC[16] were used. The hyperparameters for BAR[15] were tuned according to Table 6. We note that the default σ hyperparameter is 1 as stated in the BAR repository. However, we found that the resulting AUC Top-10 was much lower than all σ values in 6. Thus, we performed hyperparameter tuning using much larger σ values according to the values REINVENT was tuned with.

H Practical Molecular Optimization (PMO) Augmented Memory and BAR Prior

Augmented Memory required training a Prior and follows the protocol from REINVENT[13] and using the provided ZINC[62] dataset in the PMO[11] repository. Table 7 shows the hyperparameters of the LSTM[45] network. We note all hyperparameters were kept default and the model was trained for 10 epochs as SMILES validity reached 95% and the total wall time was 11 minutes 57 seconds. BAR[15] experiments were run with this same pre-trained Prior.

Table 6: Best Agent Reminder (BAR)[15] hyperparameter tuning for Experiment 2: Practical Molecular Optimization (PMO)[11]. The AUC top-10 was used to assess performance and was based on the protocol proposed in the PMO benchmark: average AUC top-10 across 3 independent runs of zaleplon_mpo and perindopril_mpo.

Sigma (σ)	Alpha (α)	Top-10 AUC
250	0.25	0.610
250	0.50	0.677
500	0.25	0.708
500	0.50	0.728
750	0.25	0.739
750	0.50	0.732
1000	**0.25**	**0.762**
1000	0.50	0.759

Table 7: LSTM model hyperparameters for Augmented Memory and BAR

Cell Type	LSTM
Number of Layers	3
Embedding Layer Size	256
Dropout	0
Training Batch Size	128
SMILES Training Randomization	True

I DRD2 Experiment Wall Times

Table 8: Experiment 3: Dopamine Type 2 Receptor (DRD2) Case Study wall times.

Algorithm	Wall Time
Augmented Memory	21 hours 25 minutes $\pm$ 2 hours 20 minutes
REINVENT	27 hours 7 minutes $\pm$ 2 hours 21 minutes
Augmented Hill-Climbing (AHC)	30 hours 24 minutes $\pm$ 4 hours 48 minutes
Best Agent Reminder (BAR)	35 hours 18 minutes $\pm$ 2 hours 12 minutes

The wall times for Experiment 3: Dopamine Type 2 Receptor (DRD2) are presented in Table 8. We note that we performed a total of 6 replicates for each algorithm: 3 in the main result experiments and 3 in the exploiting AutoDock Vina experiments (Figure 6). For REINVENT[12,13], AHC[16], and BAR[15], we pool the experiments using experience replay. For example, REINVENT values are reported based on 12 total runs: 3 for main result experiments, 3 for main results experiments with experience replay, 3 for exploiting AutoDock Vina experiments, and 3 for exploiting AutoDock Vina experiments with experience replay. The bottleneck in all experiments is AutoDock Vina[53] and the wall time is highly variable, depending on the molecules sampled by the Agent. Finally, we note that all experiments were run with a batch size of 64 for 150 epochs. The exception is BAR which was run for 75 epochs as each epoch samples 2 batches of molecules: one from the current Agent and one from the best Agent. All experiments had an AutoDock Vina oracle budget of 9,600 calls. Finally, we comment on the variable wall times of each algorithm despite having a fixed oracle budget. There are two sources of stochasticity. Firstly, the experiments were performed on a shared cluster and compute speed is variable depending on usage. Secondly, docking is itself stochastic and generally requires more search time for larger molecules. Augmented Memory jointly optimizes for Vina, QED[57], and MW which generally enforces smaller molecules and could be a reason for the faster average compute time.

J AutoDock Vina DRD2 Receptor Preparation and Docking

The receptor grid for AutoDock Vina[53] docking against DRD2 (PDB ID: 6CM4[52]) was performed using DockStream[56]. The PDB file for 6CM4 was first downloaded from the Protein Data Bank. One monomer unit was extracted and refined using PDBFixer[63] through the DockStream wrapper. The prepared grid was centered at (x, y, z) = (9.93, 5.85, -9.58) with a search box of 15Å x 15Å x 15Å. Docking for all experiments were performed with DockStream using the following protocol: embed sampled SMILES with RDKit Universal Force Field (UFF)[64] with 600 maximum convergence iterations and execute AutoDock Vina docking parallelized over 36 CPU cores (Intel(R) Xeon(R) Platinum 8360Y processors).

K Proof of Loss Function and Policy Gradient Equivalency

In this section, we show that the loss function used to tune the Agent is equivalent to optimizing the expected reward of the policy following the REINFORCE[39] algorithm. Molecules are represented as a sequence of tokens given by the Simplified Molecular Input Line Entry System (SMILES)[14] format and generated in an autoregressive manner. The generative process is Markovian (Equation 8):

$$P(x) = \prod_{t=1}^{T} P(s_t \mid s_{t-1}, s_{t-2}, \ldots, s_1) \tag{8}$$

Equation 1 states that the probability of generating a given SMILES, x, is equal to the product of the probabilities of generating a token at time-step t, given the sequence so far at time-step $t-1$. The model is pre-trained on a dataset of molecules (ChEMBL[61] for the main experiments and ZINC[62] for the benchmarking experiment) to yield the Prior which is parameterized by the weights θ. The Agent is initialized identical to the Prior but is fine-tuned during the reinforcement learning (RL) process. The Augmented Likelihood is defined as a linear combination between the Prior and a reward term (Equation 9):

$$\log \pi_{\theta_{\text{Augmented}}} = \log \pi_{\theta_{\text{Prior}}} + \sigma S(x) \tag{9}$$

S is the reward function assessing the desirability of a sampled molecule and σ is a hyperparameter that scales the reward. A higher σ places a greater contribution on the reward function and less on the Prior. The Prior is used to ensure generated SMILES are syntactically correct and has been empirically shown to enforce reasonable chemistry. The loss function is defined as the squared difference between the Augmented Likelihood and the Agent Likelihood for a given batch, B, of sampled SMILES constructed following the actions, $a \in A^*$ (Equation 10):

$$L(\theta) = \frac{1}{|B|} \left[\sum_{a \in A^*} (\log \pi_{\theta_{\text{Augmented}}} - \log \pi_{\theta_{\text{Agent}}}) \right]^2 \tag{10}$$

Taking the derivative with respect to θ (Equation 11):

$$\nabla_\theta L(\theta) = -2 \frac{1}{|B|} \left[\sum_{a \in A^*} \log \pi_{\theta_{\text{Augmented}}} - \log \pi_{\theta_{\text{Agent}}} \right] \sum_{a \in A^*} \nabla_\theta \log \pi_{\theta_{\text{Agent}}} \tag{11}$$

Minimizing $J(\theta)$ tunes the Agent to generate molecules satisfying the reward function.

Following Fialková et al.[46], we now show that minimizing $J(\theta)$ is equivalent to optimizing the expected reward of the policy. The generative process is cast as an on-policy RL problem by defining the state space, S_t, and the action space, $A_t(s_t)$. S_t denotes every intermediate sequence of tokens leading up to the fully constructed SMILES and $A_t(s_t)$ are the token sampling probabilities at every intermediate state. $A_t(s_t)$ is controlled by the policy, π_θ, which is parameterized by the weights, θ,

of the neural network. Given a reward function, R, the objective is to maximize the expected reward when taking actions defined by the policy (Equation 12):

$$J(\theta) = \mathbb{E}_{a_t \sim \pi_\theta} \left[\sum_{t=0}^{T} R(a_t, s_t) \right] \tag{12}$$

Rewriting the expectation (Equation 13):

$$J(\theta) = \sum_{t=0}^{T} \sum_{a \in A_t} R(a_t, s_t) \pi_\theta(a_t | s_t) \tag{13}$$

The expectation can be rewritten as a double summation over all time-steps and actions taken at each time-step, following the policy, π_θ. Next, the derivative of the expression is taken (Equation 14):

$$\nabla_\theta J(\theta) = \sum_{t=0}^{T} \sum_{a \in A_t} R(a_t, s_t) \nabla_\theta \pi_\theta(a_t | s_t) \tag{14}$$

Applying the log-derivative trick (Equation 15):

$$\nabla_\theta J(\theta) = \sum_{t=0}^{T} \sum_{a \in A_t} R(a_t, s_t) \pi_\theta(a_t | s_t) \nabla_\theta \log \pi_\theta(a_t | s_t) \tag{15}$$

Using the definition of expectation for discrete variables, i.e., the policy actions which can only sample the vocabulary tokens (Equation 16):

$$\nabla_\theta J(\theta) = \mathbb{E}_{a_t \sim \pi_\theta} \left[\sum_{t=0}^{T} R(a_t, s_t) \nabla_\theta \log \pi_\theta(a_t | s_t) \right] \tag{16}$$

As computing the expectation is intractable, it is instead approximated by sampling a batch, B, of trajectories, i.e., SMILES strings. The process of SMILES generation is further defined as an episodic task where reward is only given at the terminal state. In particular, the *desirability* of a SMILES sequence only applies once the full SMILES string has been sampled and it maps to a valid molecule. Thus, all intermediate rewards are 0. Defining the set of actions taken in a batch, A^* as the specific token sequences generated at a given epoch yields Equation 17:

$$\nabla_\theta J(\theta) = \frac{1}{|B|} \left[\sum_{t=0}^{T} \sum_{a \in A^*} R(a_t, s_t) \nabla_\theta \log \pi_\theta(a_t | s_t) \right] \tag{17}$$

Finally, the reward, R is defined according to Fialková et al.[46] (Equation 18):

$$R(a_t, s_t) = \log \pi_{\theta_{\text{Augmented}}} - \log \pi_{\theta_{\text{Agent}}} \tag{18}$$

Substituting Equation 18 into Equation 17 yields the desired equivalency to the loss function (Equation 11) up to a constant factor.

L Augmented Memory Algorithm

The pseudo-code for Augmented Memory is presented here.

Algorithm 1: Augmented Memory

Input: Prior π_{Prior}, Epochs N, Augmentation Rounds A, Scoring Function S, Sigma σ
Output: Fine-tuned Agent Policy $\pi_{\theta_{\text{Agent}}}$, Generated Molecules G

Initialization:
Generative Agent $\pi_{\theta_{\text{Agent}}} = \pi_{\text{Prior}}$;
Diversity Filter DF;
Replay Buffer $B = \{\}$;
for $i \leftarrow 1$ **to** N **do**
  Sample batch of SMILES $X = \{x_1, \ldots, x_b\}$ with $x_i \sim \pi_{\theta_{\text{Agent}}}$;
  Compute reward using the scoring function $S(X)$;
  Modify reward based on the diversity filter $DF(S(X))$;
  Update replay buffer $B_i = X_i \cup X_{i-1}$;
  (Optionally) purge replay buffer;
  Compute Augmented Likelihood $\log \pi_{\theta_{\text{Augmented}}} = \log \pi_{\text{Prior}}(X) + \sigma S(X)$;
  Compute loss $J(\theta) = (\log \pi_{\text{Augmented}} - \log \pi_{\theta_{\text{Agent}}}(X))^2$;
  Update the Agent's policy $\pi_{\theta_{\text{Agent}}}$;
  for $j \leftarrow 1$ **to** A **do**
    Augment sampled SMILES $X_{\text{Augmented}}$;
    Compute Augmented Likelihood of augmented SMILES (reward is unchanged) $\log \pi_{\text{Augmented}} = \log \pi_{\text{Prior}}(X_{\text{Augmented}}) + \sigma S(X)$;
    Compute loss $J(\theta)_{\text{Augmented}} = (\log \pi_{\text{Augmented}} - \log \pi_{\theta_{\text{Agent}}}(X_{\text{Augmented}}))^2$;
    Augment entire replay buffer $B_{\text{Augmented}}$;
    Compute Augmented Likelihood on the augmented buffer (reward is the buffer stored rewards) $\log \pi_{\text{Buffer Augmented}} = \log \pi_{\text{Prior}}(B_{\text{Augmented}}) + \sigma S(B)$;
    Compute augmented buffer loss $J(\theta)_{\text{Buffer Augmented}} = (\log \pi_{\text{Buffer Augmented}} - \log \pi_{\theta_{\text{Agent}}}(B_{\text{Augmented}}))^2$;
    Concatenate the augmented sampled SMILES loss and the augmented buffer loss $J(\theta)_{\text{Augmented Memory}} = J(\theta)_{\text{Augmented}} + J(\theta)_{\text{Buffer Augmented}}$;
    Update the Agent's policy $\pi_{\theta_{\text{Agent}}}$;

References

1. Benjamin Sanchez-Lengeling and Alán Aspuru-Guzik. Inverse molecular design using machine learning: Generative models for matter engineering. *Science*, 361(6400):360–365, July 2018. doi: 10.1126/science.aat2663. URL `https://www.science.org/doi/10.1126/science.aat2663`. Publisher: American Association for the Advancement of Science.

2. Julia Westermayr, Joe Gilkes, Rhyan Barrett, and Reinhard J. Maurer. High-throughput property-driven generative design of functional organic molecules. *Nat Comput Sci*, 3(2):139–148, February 2023. ISSN 2662-8457. doi: 10.1038/s43588-022-00391-1. URL `https://www.nature.com/articles/s43588-022-00391-1`. Number: 2 Publisher: Nature Publishing Group.

3. Jiankun Lyu, Sheng Wang, Trent E Balius, Isha Singh, Anat Levit, Yurii S Moroz, Matthew J O'Meara, Tao Che, Enkhjargal Algaa, Kateryna Tolmachova, et al. Ultra-large library docking for discovering new chemotypes. *Nature*, 566(7743):224–229, 2019.

4. Alex Zhavoronkov, Yan A. Ivanenkov, Alex Aliper, Mark S. Veselov, Vladimir A. Aladinskiy, Anastasiya V. Aladinskaya, Victor A. Terentiev, Daniil A. Polykovskiy, Maksim D. Kuznetsov, Arip Asadulaev, Yury Volkov, Artem Zholus, Rim R. Shayakhmetov, Alexander Zhebrak,

Lidiya I. Minaeva, Bogdan A. Zagribelnyy, Lennart H. Lee, Richard Soll, David Madge, Li Xing, Tao Guo, and Alán Aspuru-Guzik. Deep learning enables rapid identification of potent DDR1 kinase inhibitors. *Nat Biotechnol*, 37(9):1038–1040, September 2019. ISSN 1546-1696. doi: 10.1038/s41587-019-0224-x. URL https://www.nature.com/articles/s41587-019-0224-x. Number: 9 Publisher: Nature Publishing Group.

5. Feng Ren, Xiao Ding, Min Zheng, Mikhail Korzinkin, Xin Cai, Wei Zhu, Alexey Mantsyzov, Alex Aliper, Vladimir Aladinskiy, Zhongying Cao, Shanshan Kong, Xi Long, Bonnie Hei Man Liu, Yingtao Liu, Vladimir Naumov, Anastasia Shneyderman, Ivan V. Ozerov, Ju Wang, Frank W. Pun, Daniil A. Polykovskiy, Chong Sun, Michael Levitt, Alán Aspuru-Guzik, and Alex Zhavoronkov. AlphaFold accelerates artificial intelligence powered drug discovery: efficient discovery of a novel CDK20 small molecule inhibitor. *Chemical Science*, 14(6):1443–1452, 2023. doi: 10.1039/D2SC05709C. URL https://pubs.rsc.org/en/content/articlelanding/2023/sc/d2sc05709c. Publisher: Royal Society of Chemistry.

6. Julius Seumer, Jonathan Kirschner Solberg Hansen, Mogens Brøndsted Nielsen, and Jan H Jensen. Computational evolution of new catalysts for the morita–baylis–hillman reaction. *Angewandte Chemie International Edition*, page e202218565, 2022.

7. Arman A. Sadybekov, Anastasiia V. Sadybekov, Yongfeng Liu, Christos Iliopoulos-Tsoutsouvas, Xi-Ping Huang, Julie Pickett, Blake Houser, Nilkanth Patel, Ngan K. Tran, Fei Tong, Nikolai Zvonok, Manish K. Jain, Olena Savych, Dmytro S. Radchenko, Spyros P. Nikas, Nicos A. Petasis, Yurii S. Moroz, Bryan L. Roth, Alexandros Makriyannis, and Vsevolod Katritch. Synthon-based ligand discovery in virtual libraries of over 11 billion compounds. *Nature*, 601(7893):452–459, January 2022. ISSN 1476-4687. doi: 10.1038/s41586-021-04220-9. URL https://www.nature.com/articles/s41586-021-04220-9. Number: 7893 Publisher: Nature Publishing Group.

8. Joshua Meyers, Benedek Fabian, and Nathan Brown. De novo molecular design and generative models. *Drug Discovery Today*, 26(11):2707–2715, November 2021. ISSN 1359-6446. doi: 10.1016/j.drudis.2021.05.019. URL https://www.sciencedirect.com/science/article/pii/S1359644621002531.

9. Atsushi Yoshimori, Yasunobu Asawa, Enzo Kawasaki, Tomohiko Tasaka, Seiji Matsuda, Toru Sekikawa, Satoshi Tanabe, Masahiro Neya, Hideaki Natsugari, and Chisato Kanai. Design and Synthesis of DDR1 Inhibitors with a Desired Pharmacophore Using Deep Generative Models. *ChemMedChem*, 16(6):955–958, 2021. ISSN 1860-7187. doi: 10.1002/cmdc.202000786. URL https://onlinelibrary.wiley.com/doi/abs/10.1002/cmdc.202000786. _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/cmdc.202000786.

10. Maria Korshunova, Niles Huang, Stephen Capuzzi, Dmytro S. Radchenko, Olena Savych, Yuriy S. Moroz, Carrow I. Wells, Timothy M. Willson, Alexander Tropsha, and Olexandr Isayev. Generative and reinforcement learning approaches for the automated de novo design of bioactive compounds. *Commun Chem*, 5(1):1–11, October 2022. ISSN 2399-3669. doi: 10.1038/s42004-022-00733-0. URL https://www.nature.com/articles/s42004-022-00733-0. Number: 1 Publisher: Nature Publishing Group.

11. Wenhao Gao, Tianfan Fu, Jimeng Sun, and Connor W. Coley. Sample Efficiency Matters: A Benchmark for Practical Molecular Optimization, October 2022. URL http://arxiv.org/abs/2206.12411. arXiv:2206.12411 [cs, q-bio].

12. Marcus Olivecrona, Thomas Blaschke, Ola Engkvist, and Hongming Chen. Molecular de-novo design through deep reinforcement learning. *Journal of Cheminformatics*, 9(1):48, September 2017. ISSN 1758-2946. doi: 10.1186/s13321-017-0235-x. URL https://doi.org/10.1186/s13321-017-0235-x.

13. Thomas Blaschke, Josep Arús-Pous, Hongming Chen, Christian Margreitter, Christian Tyrchan, Ola Engkvist, Kostas Papadopoulos, and Atanas Patronov. REINVENT 2.0: An AI Tool for De Novo Drug Design. *J. Chem. Inf. Model.*, 60(12):5918–5922, December 2020. ISSN 1549-9596. doi: 10.1021/acs.jcim.0c00915. URL https://doi.org/10.1021/acs.jcim.0c00915. Publisher: American Chemical Society.

14. David Weininger. SMILES, a chemical language and information system. 1. Introduction to methodology and encoding rules. *J. Chem. Inf. Comput. Sci.*, 28(1):31–36, February 1988. ISSN 0095-2338. doi: 10.1021/ci00057a005. URL https://doi.org/10.1021/ci00057a005. Publisher: American Chemical Society.

15. Sara Romeo Atance, Juan Viguera Diez, Ola Engkvist, Simon Olsson, and Rocío Mercado. De novo drug design using reinforcement learning with graph-based deep generative models. *Journal of Chemical Information and Modeling*, 62(20):4863–4872, 2022.

16. Morgan Thomas, Noel M. O'Boyle, Andreas Bender, and Chris de Graaf. Augmented Hill-Climb increases reinforcement learning efficiency for language-based de novo molecule generation. *Journal of Cheminformatics*, 14(1):68, October 2022. ISSN 1758-2946. doi: 10.1186/s13321-022-00646-z. URL https://doi.org/10.1186/s13321-022-00646-z.

17. Esben Jannik Bjerrum, Christian Margreitter, Thomas Blaschke, and Raquel López-Ríos de Castro. Faster and more diverse de novo molecular optimization with double-loop reinforcement learning using augmented SMILES, March 2023. URL http://arxiv.org/abs/2210.12458. arXiv:2210.12458 [physics].

18. Morgan Thomas, Noel M O'Boyle, Andreas Bender, and Chris De Graaf. Re-evaluating sample efficiency in de novo molecule generation. *arXiv preprint arXiv:2212.01385*, 2022.

19. Mariya Popova, Olexandr Isayev, and Alexander Tropsha. Deep reinforcement learning for de novo drug design. *Science Advances*, 4(7):eaap7885, July 2018. doi: 10.1126/sciadv.aap7885. URL https://www.science.org/doi/10.1126/sciadv.aap7885. Publisher: American Association for the Advancement of Science.

20. Marwin HS Segler, Thierry Kogej, Christian Tyrchan, and Mark P Waller. Generating focused molecule libraries for drug discovery with recurrent neural networks. *ACS central science*, 4(1): 120–131, 2018.

21. Niclas Ståhl, Goran Falkman, Alexander Karlsson, Gunnar Mathiason, and Jonas Bostrom. Deep reinforcement learning for multiparameter optimization in de novo drug design. *Journal of chemical information and modeling*, 59(7):3166–3176, 2019.

22. Manan Goel, Shampa Raghunathan, Siddhartha Laghuvarapu, and U Deva Priyakumar. Molegular: molecule generation using reinforcement learning with alternating rewards. *Journal of Chemical Information and Modeling*, 61(12):5815–5826, 2021.

23. Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative Adversarial Networks, June 2014. URL http://arxiv.org/abs/1406.2661. arXiv:1406.2661 [cs, stat].

24. Benjamin Sanchez-Lengeling, Carlos Outeiral, Gabriel L Guimaraes, and Alan Aspuru-Guzik. Optimizing distributions over molecular space. an objective-reinforced generative adversarial network for inverse-design chemistry (organic). 2017.

25. Evgeny Putin, Arip Asadulaev, Yan Ivanenkov, Vladimir Aladinskiy, Benjamin Sanchez-Lengeling, Alán Aspuru-Guzik, and Alex Zhavoronkov. Reinforced Adversarial Neural Computer for de Novo Molecular Design. *J. Chem. Inf. Model.*, 58(6):1194–1204, June 2018. ISSN 1549-9596. doi: 10.1021/acs.jcim.7b00690. URL https://doi.org/10.1021/acs.jcim.7b00690. Publisher: American Chemical Society.

26. Gabriel Lima Guimaraes, Benjamin Sanchez-Lengeling, Carlos Outeiral, Pedro Luis Cunha Farias, and Alán Aspuru-Guzik. Objective-Reinforced Generative Adversarial Networks (ORGAN) for Sequence Generation Models, February 2018. URL http://arxiv.org/abs/1705.10843. arXiv:1705.10843 [cs, stat].

27. Nicola De Cao and Thomas Kipf. MolGAN: An implicit generative model for small molecular graphs, September 2022. URL http://arxiv.org/abs/1805.11973. arXiv:1805.11973 [cs, stat].

28. Diederik P. Kingma and Max Welling. Auto-Encoding Variational Bayes, December 2022. URL http://arxiv.org/abs/1312.6114. arXiv:1312.6114 [cs, stat].

29. Rafael Gómez-Bombarelli, Jennifer N Wei, David Duvenaud, José Miguel Hernández-Lobato, Benjamín Sánchez-Lengeling, Dennis Sheberla, Jorge Aguilera-Iparraguirre, Timothy D Hirzel, Ryan P Adams, and Alán Aspuru-Guzik. Automatic chemical design using a data-driven continuous representation of molecules. *ACS central science*, 4(2):268–276, 2018.

30. Jiaxuan You, Bowen Liu, Rex Ying, Vijay Pande, and Jure Leskovec. Graph Convolutional Policy Network for Goal-Directed Molecular Graph Generation, February 2019. URL http://arxiv.org/abs/1806.02473. arXiv:1806.02473 [cs, stat].

31. Wengong Jin, Dr Regina Barzilay, and Tommi Jaakkola. Multi-Objective Molecule Generation using Interpretable Substructures. In *Proceedings of the 37th International Conference on Machine Learning*, pages 4849–4859. PMLR, November 2020. URL https://proceedings.mlr.press/v119/jin20b.html. ISSN: 2640-3498.

32. Rocío Mercado, Tobias Rastemo, Edvard Lindelöf, Günter Klambauer, Ola Engkvist, Hongming Chen, and Esben Jannik Bjerrum. Graph networks for molecular design. *Mach. Learn.: Sci. Technol.*, 2(2):025023, March 2021. ISSN 2632-2153. doi: 10.1088/2632-2153/abcf91. URL https://dx.doi.org/10.1088/2632-2153/abcf91. Publisher: IOP Publishing.

33. Emmanuel Bengio, Moksh Jain, Maksym Korablyov, Doina Precup, and Yoshua Bengio. Flow network based generative models for non-iterative diverse candidate generation. *Advances in Neural Information Processing Systems*, 34:27381–27394, 2021.

34. Tianfan Fu, Wenhao Gao, Connor Coley, and Jimeng Sun. Reinforced genetic algorithm for structure-based drug design. *Advances in Neural Information Processing Systems*, 35:12325–12338, 2022.

35. Zhenpeng Zhou, Steven Kearnes, Li Li, Richard N Zare, and Patrick Riley. Optimization of molecules via deep reinforcement learning. *Scientific reports*, 9(1):1–10, 2019.

36. Daniel Flam-Shepherd, Kevin Zhu, and Alán Aspuru-Guzik. Language models can learn complex molecular distributions. *Nature Communications*, 13(1):3293, 2022.

37. Nathan Brown, Marco Fiscato, Marwin H.S. Segler, and Alain C. Vaucher. GuacaMol: Benchmarking Models for de Novo Molecular Design. *J. Chem. Inf. Model.*, 59(3):1096–1108, March 2019. ISSN 1549-9596, 1549-960X. doi: 10.1021/acs.jcim.8b00839. URL https://pubs.acs.org/doi/10.1021/acs.jcim.8b00839.

38. Daniil Polykovskiy, Alexander Zhebrak, Benjamin Sanchez-Lengeling, Sergey Golovanov, Oktai Tatanov, Stanislav Belyaev, Rauf Kurbanov, Aleksey Artamonov, Vladimir Aladinskiy, Mark Veselov, Artur Kadurin, Simon Johansson, Hongming Chen, Sergey Nikolenko, Alán Aspuru-Guzik, and Alex Zhavoronkov. Molecular Sets (MOSES): A Benchmarking Platform for Molecular Generation Models. *Frontiers in Pharmacology*, 11, 2020. ISSN 1663-9812. URL https://www.frontiersin.org/articles/10.3389/fphar.2020.565644.

39. Ronald J. Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Mach Learn*, 8(3):229–256, May 1992. ISSN 1573-0565. doi: 10.1007/BF00992696. URL https://doi.org/10.1007/BF00992696.

40. Daniel Neil, Marwin Segler, Laura Guasch, Mohamed Ahmed, Dean Plumbley, Matthew Sellwood, and Nathan Brown. EXPLORING DEEP RECURRENT MODELS WITH REIN- FORCEMENT LEARNING FOR MOLECULE DESIGN. 2018.

41. Long-Ji Lin. Self-improving reactive agents based on reinforcement learning, planning and teaching.

42. William Fedus, Prajit Ramachandran, Rishabh Agarwal, Yoshua Bengio, Hugo Larochelle, Mark Rowland, and Will Dabney. Revisiting Fundamentals of Experience Replay, July 2020. URL http://arxiv.org/abs/2007.06700. arXiv:2007.06700 [cs, stat].

43. Thomas Blaschke, Ola Engkvist, Jürgen Bajorath, and Hongming Chen. Memory-assisted reinforcement learning for diverse molecular de novo design. *Journal of Cheminformatics*, 12 (1):68, November 2020. ISSN 1758-2946. doi: 10.1186/s13321-020-00473-0. URL https://doi.org/10.1186/s13321-020-00473-0.

44. Eric Wiewiora. Reward Shaping. In Claude Sammut and Geoffrey I. Webb, editors, *Encyclopedia of Machine Learning*, pages 863–865. Springer US, Boston, MA, 2010. ISBN 978-0-387-30164-8. doi: 10.1007/978-0-387-30164-8_731. URL https://doi.org/10.1007/978-0-387-30164-8_731.

45. Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8): 1735–1780, 1997.

46. Vendy Fialková, Jiaxi Zhao, Kostas Papadopoulos, Ola Engkvist, Esben Jannik Bjerrum, Thierry Kogej, and Atanas Patronov. LibINVENT: Reaction-based Generative Scaffold Decoration for in Silico Library Design. *J. Chem. Inf. Model.*, 62(9):2046–2063, May 2022. ISSN 1549-9596. doi: 10.1021/acs.jcim.1c00469. URL https://doi.org/10.1021/acs.jcim.1c00469. Publisher: American Chemical Society.

47. Michael Moret, Lukas Friedrich, Francesca Grisoni, Daniel Merk, and Gisbert Schneider. Generative molecular design in low data regimes. *Nat Mach Intell*, 2(3):171–180, March 2020. ISSN 2522-5839. doi: 10.1038/s42256-020-0160-y. URL https://www.nature.com/articles/s42256-020-0160-y. Number: 3 Publisher: Nature Publishing Group.

48. Josep Arús-Pous, Simon Viet Johansson, Oleksii Prykhodko, Esben Jannik Bjerrum, Christian Tyrchan, Jean-Louis Reymond, Hongming Chen, and Ola Engkvist. Randomized SMILES strings improve the quality of molecular generative models. *Journal of Cheminformatics*, 11 (1):71, November 2019. ISSN 1758-2946. doi: 10.1186/s13321-019-0393-0. URL https://doi.org/10.1186/s13321-019-0393-0.

49. Leland McInnes, John Healy, and James Melville. UMAP: Uniform Manifold Approximation and Projection for Dimension Reduction, September 2020. URL http://arxiv.org/abs/1802.03426. arXiv:1802.03426 [cs, stat].

50. David Rogers and Mathew Hahn. Extended-connectivity fingerprints. *Journal of chemical information and modeling*, 50(5):742–754, 2010.

51. Chemcharts. *https://github.com/SMargreitter/ChemCharts*.

52. Sheng Wang, Tao Che, Anat Levit, Brian K Shoichet, Daniel Wacker, and Bryan L Roth. Structure of the d2 dopamine receptor bound to the atypical antipsychotic drug risperidone. *Nature*, 555 (7695):269–273, 2018.

53. Oleg Trott and Arthur J Olson. Autodock vina: improving the speed and accuracy of docking with a new scoring function, efficient optimization, and multithreading. *Journal of computational chemistry*, 31(2):455–461, 2010.

54. John A Arnott and Sonia Lobo Planey. The influence of lipophilicity in drug discovery and design. *Expert opinion on drug discovery*, 7(10):863–875, 2012.

55. AkshatKumar Nigam, Robert Pollice, and Alán Aspuru-Guzik. Parallel tempered genetic algorithm guided by deep neural networks for inverse molecular design. *Digital Discovery*, 1(4): 390–404, 2022.

56. Jeff Guo, Jon Paul Janet, Matthias R. Bauer, Eva Nittinger, Kathryn A. Giblin, Kostas Papadopoulos, Alexey Voronov, Atanas Patronov, Ola Engkvist, and Christian Margreitter. DockStream: a docking wrapper to enhance de novo molecular design. *Journal of Cheminformatics*, 13(1):89, November 2021. ISSN 1758-2946. doi: 10.1186/s13321-021-00563-7. URL https://doi.org/10.1186/s13321-021-00563-7.

57. G. Richard Bickerton, Gaia V. Paolini, Jérémy Besnard, Sorel Muresan, and Andrew L. Hopkins. Quantifying the chemical beauty of drugs. *Nature Chem*, 4(2):90–98, February 2012. ISSN 1755-4349. doi: 10.1038/nchem.1243. URL https://www.nature.com/articles/nchem.1243. Number: 2 Publisher: Nature Publishing Group.

58. AkshatKumar Nigam, Robert Pollice, Mario Krenn, Gabriel dos Passos Gomes, and Alan Aspuru-Guzik. Beyond generative models: superfast traversal, optimization, novelty, exploration and discovery (stoned) algorithm for molecules using selfies. *Chemical science*, 12(20):7079–7090, 2021.

59. Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. Curriculum learning. In *Proceedings of the 26th annual international conference on machine learning*, pages 41–48, 2009.

60. Jeff Guo, Vendy Fialková, Juan Diego Arango, Christian Margreitter, Jon Paul Janet, Kostas Papadopoulos, Ola Engkvist, and Atanas Patronov. Improving de novo molecular design with curriculum learning. *Nat Mach Intell*, 4(6):555–563, June 2022. ISSN 2522-5839. doi: 10.1038/s42256-022-00494-4. URL https://www.nature.com/articles/s42256-022-00494-4. Number: 6 Publisher: Nature Publishing Group.

61. Anna Gaulton, Louisa J. Bellis, A. Patricia Bento, Jon Chambers, Mark Davies, Anne Hersey, Yvonne Light, Shaun McGlinchey, David Michalovich, Bissan Al-Lazikani, and John P. Overington. ChEMBL: a large-scale bioactivity database for drug discovery. *Nucleic Acids Res*, 40 (Database issue):D1100–D1107, January 2012. ISSN 0305-1048. doi: 10.1093/nar/gkr777. URL https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3245175/.

62. John J. Irwin, Khanh G. Tang, Jennifer Young, Chinzorig Dandarchuluun, Benjamin R. Wong, Munkhzul Khurelbaatar, Yurii S. Moroz, John Mayfield, and Roger A. Sayle. ZINC20—A Free Ultralarge-Scale Chemical Database for Ligand Discovery. *J. Chem. Inf. Model.*, 60 (12):6065–6073, December 2020. ISSN 1549-9596. doi: 10.1021/acs.jcim.0c00675. URL https://doi.org/10.1021/acs.jcim.0c00675. Publisher: American Chemical Society.

63. Peter Eastman, Jason Swails, John D Chodera, Robert T McGibbon, Yutong Zhao, Kyle A Beauchamp, Lee-Ping Wang, Andrew C Simmonett, Matthew P Harrigan, Chaya D Stern, et al. Openmm 7: Rapid development of high performance algorithms for molecular dynamics. *PLoS computational biology*, 13(7):e1005659, 2017.

64. Anthony K Rappé, Carla J Casewit, KS Colwell, William A Goddard III, and W Mason Skiff. Uff, a full periodic table force field for molecular mechanics and molecular dynamics simulations. *Journal of the American chemical society*, 114(25):10024–10035, 1992.